Machine Learning 03

Kihyun Shin
DMSE, HBNU

TheMAD

Overview
(Cost and loss function)

TheMAD

Simplified loss function

$$L\left(f\left(x^{(i)}\right), y^{(i)}\right) = \begin{cases} -log\left(f_{w,b}\left(x^{(i)}\right)\right) & if\ y^{(i)} = 1 \\ -log\left(1 - f_{w,b}\left(x^{(i)}\right)\right) & if\ y^{(i)} = 0 \end{cases}$$

equivalent

$$L\left(f\left(x^{(i)}\right), y^{(i)}\right) = -y^{(i)} log\left(f_{w,b}\left(x^{(i)}\right)\right) - (1 - y^{(i)})\ log\left(1 - f_{w,b}\left(x^{(i)}\right)\right)$$

$if\ y^{(i)} = 1$

$$L\left(f\left(x^{(i)}\right), y^{(i)}\right) = -1 log\left(f_{w,b}\left(x^{(i)}\right)\right) - (1 - 1)\ log\left(1 - f_{w,b}\left(x^{(i)}\right)\right)$$

$if\ y^{(i)} = 0$

$$L\left(f\left(x^{(i)}\right), y^{(i)}\right) = -0 log\left(f_{w,b}\left(x^{(i)}\right)\right) - (1 - 0)\ log\left(1 - f_{w,b}\left(x^{(i)}\right)\right)$$

Simplified cost function

$$L\left(f\left(x^{(i)}\right), y^{(i)}\right) = -y^{(i)} log\left(f_{w,b}\left(x^{(i)}\right)\right) - (1-y^{(i)}) log(1 - f_{w,b}\left(x^{(i)}\right))$$

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} L\left(f\left(x^{(i)}\right), y^{(i)}\right)$$

cost — loss

$$= -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} log\left(f_{w,b}\left(x^{(i)}\right)\right) + (1-y^{(i)}) log(1 - f_{w,b}\left(x^{(i)}\right))$$

This particular cost function derived from statistics using a statistical principle called maximum likelihood estimation (idea from statistics on how to efficiently find parameters for different models)

→ Pretty much everyone uses to train logistic regression

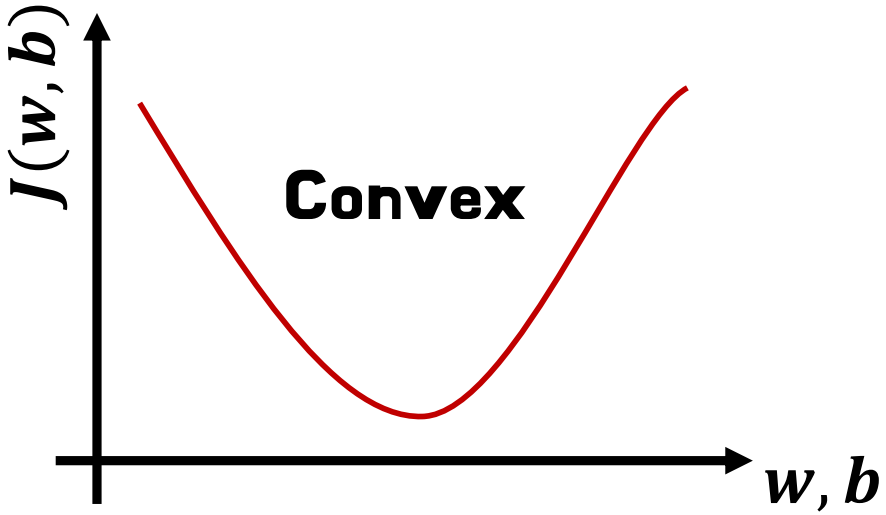
Squared error cost

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left(f(x^{(i)}) - y^{(i)}\right)^2$$

$$= L(f(x^{(i)}), y^{(i)}) \quad \textbf{Loss}$$

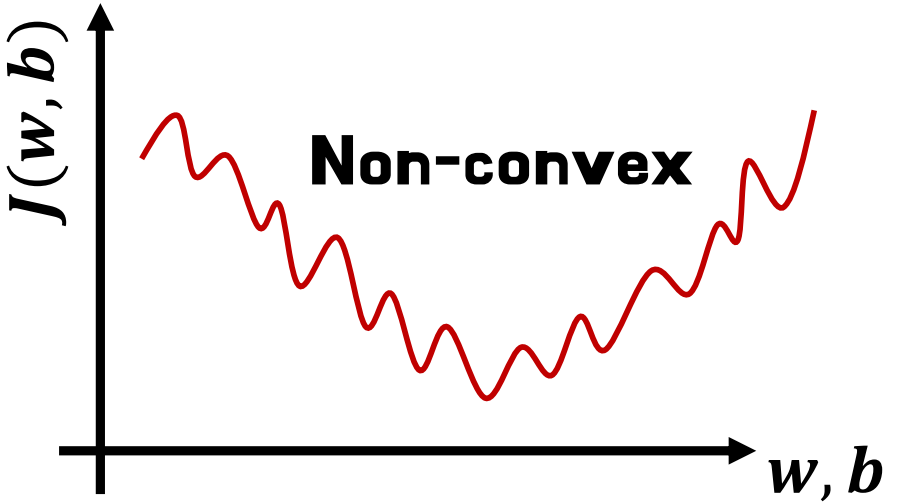
Linear regression

$$f_{w,b}(x) = wx + b$$



Logistic regression

$$f_{w,b}(x) = \frac{1}{1 + e^{-(wx+b)}}$$

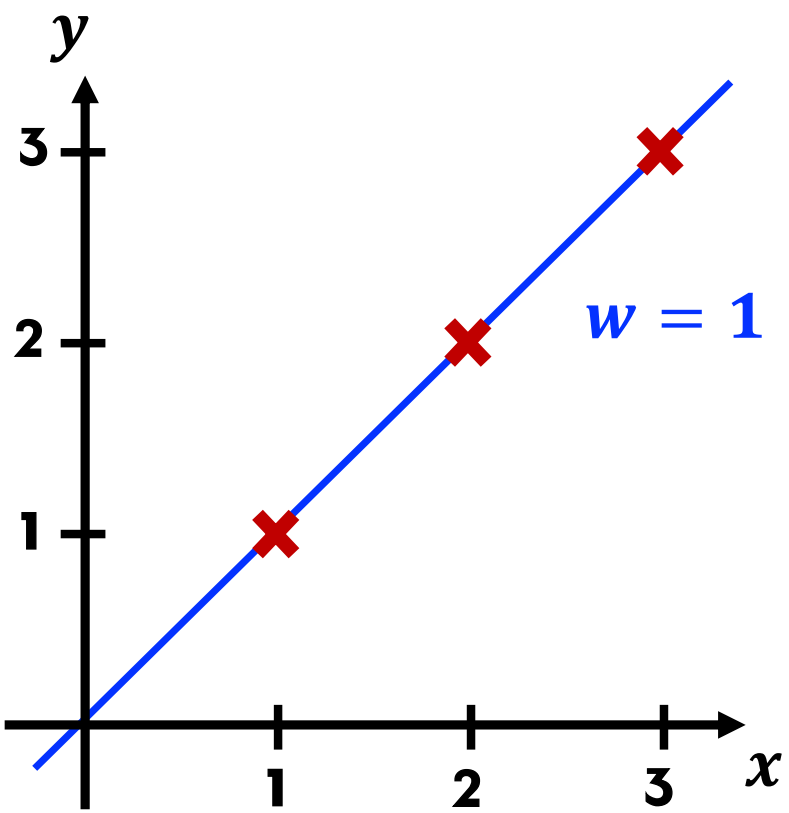


TheMAD

Cost function

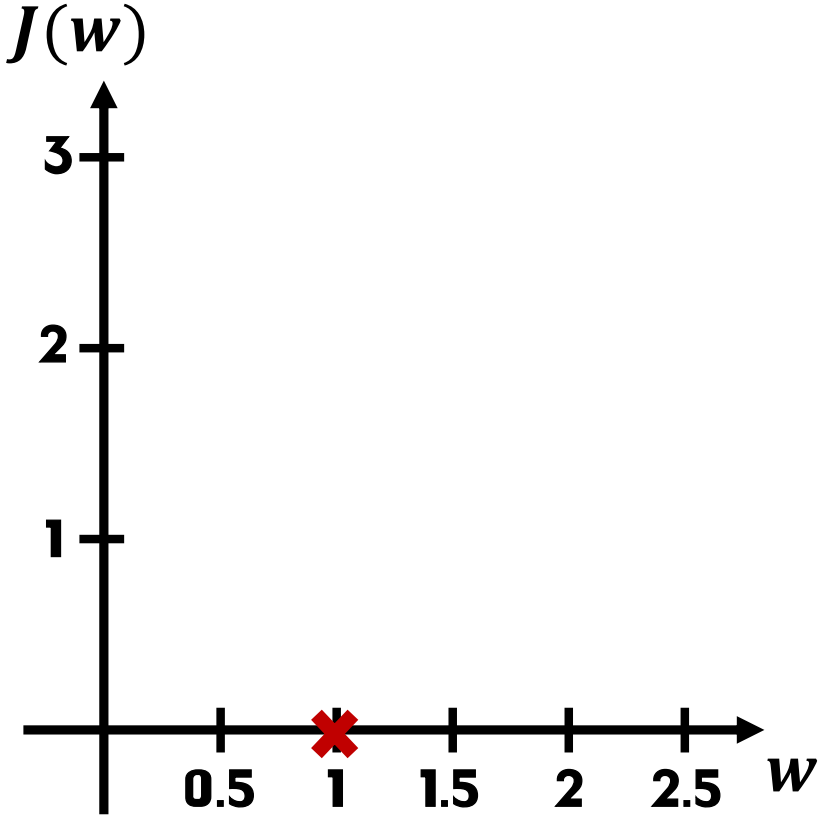
$f_w(x)$

For fixed w, function of x



$J(w)$

Function of w

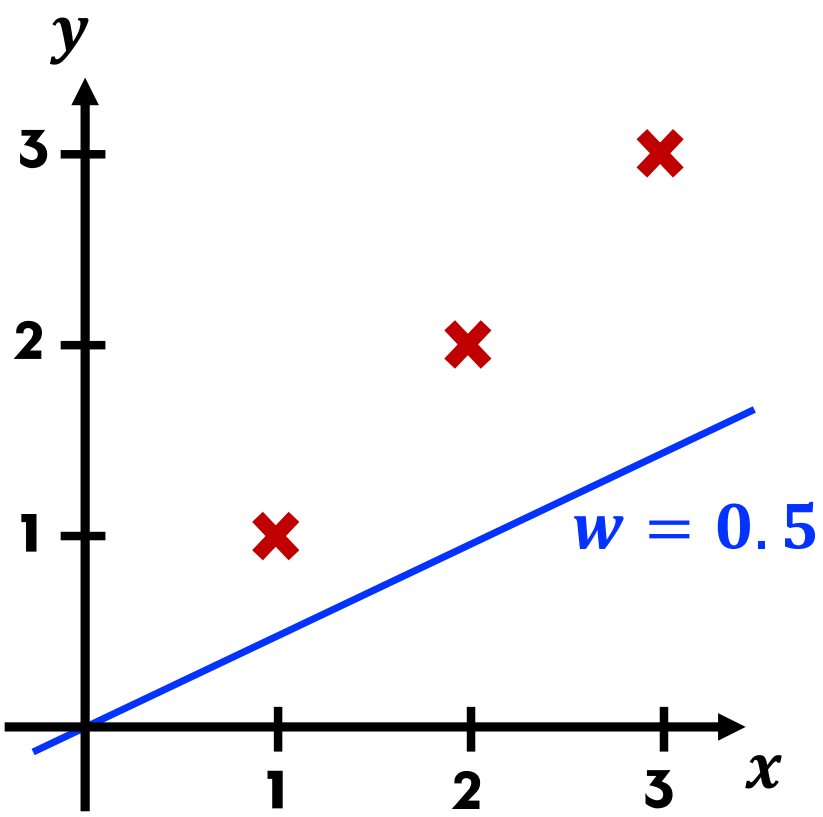


$$J(w) = \frac{1}{2m}\sum_{i=1}^{m}\left(f\left(x^{(i)}\right) - y^{(i)}\right)^2 = \frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} - y^{(i)}\right)^2 = \frac{1}{2m}\left(0^2 + 0^2 + 0^2\right) = 0$$

Cost function

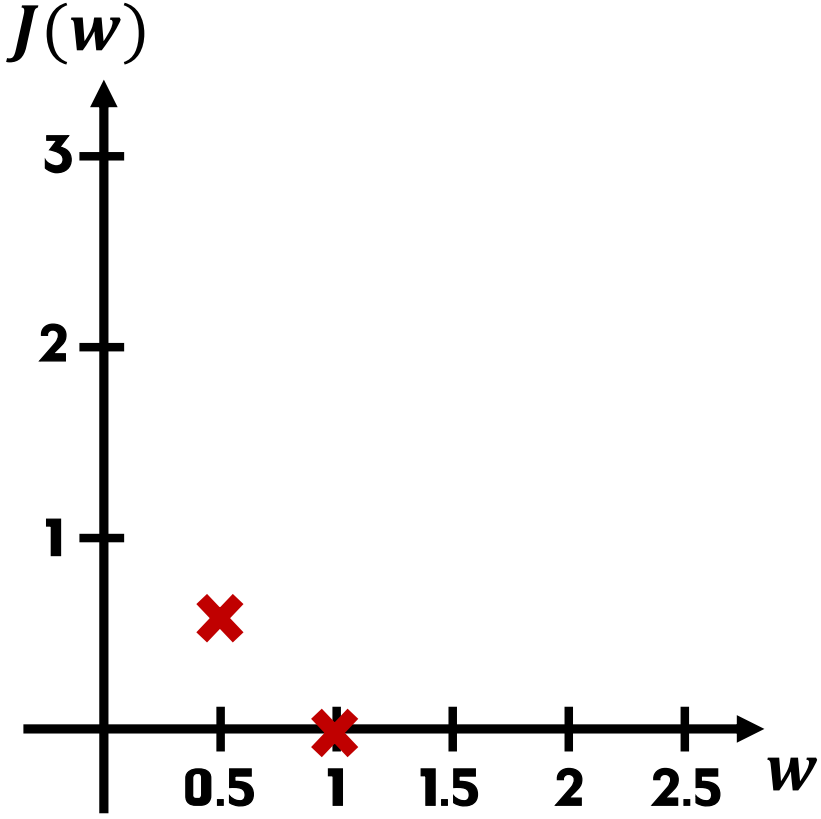
$f_w(x)$

For fixed w, function of x



$J(w)$

Function of w

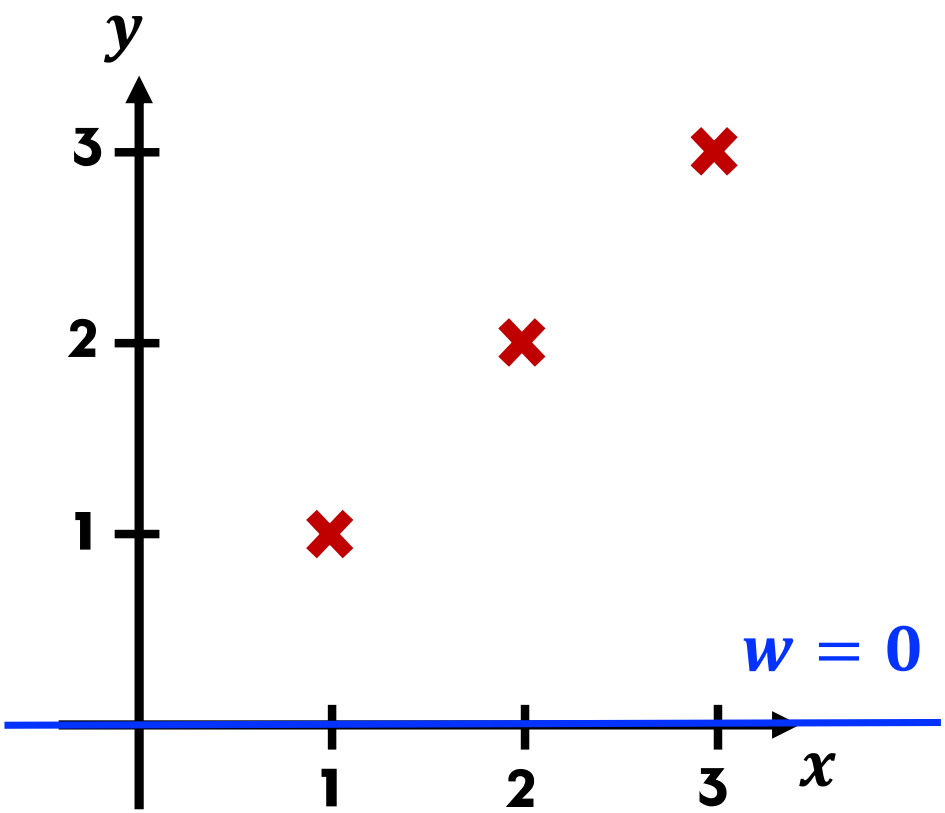


$$J(w) = \frac{1}{2m}\left((0.5-1)^2+(1-2)^2+(1.5-3)^2\right) = \frac{3.5}{6} = 0.58$$

Cost function

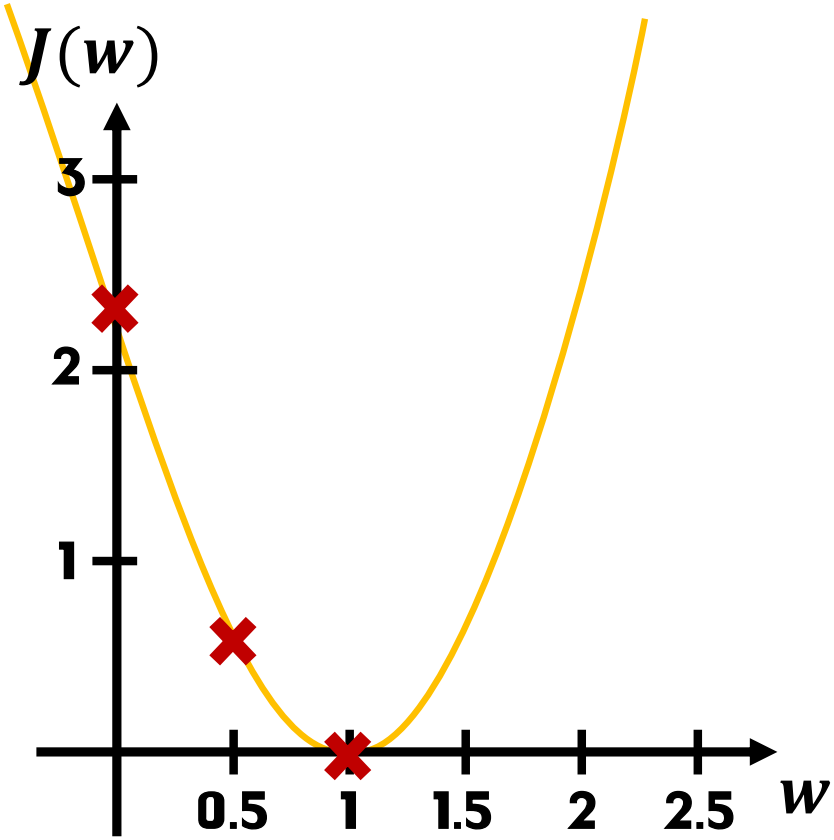
$f_w(x)$

For fixed w, function of x



$J(w)$

Function of w

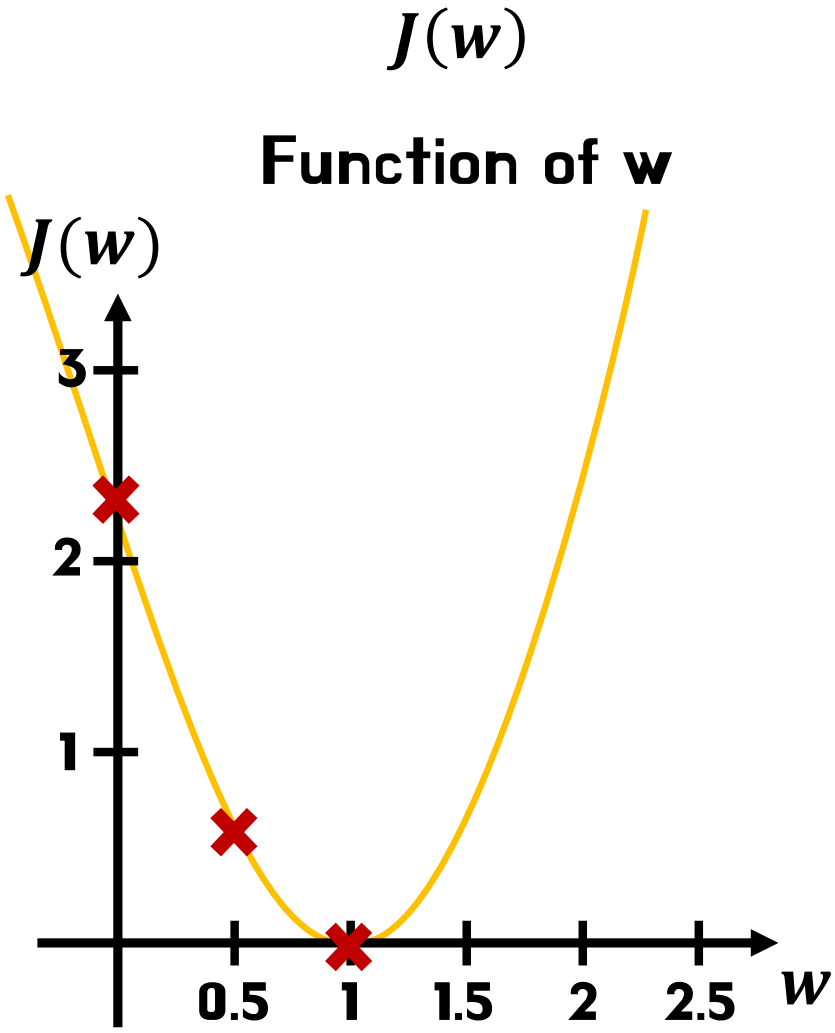


$$J(w) = \frac{1}{2m}\left((1)^2+(2)^2+(3)^2\right) = \frac{14}{6} = 2.3$$

Cost function

Goal of linear regression:

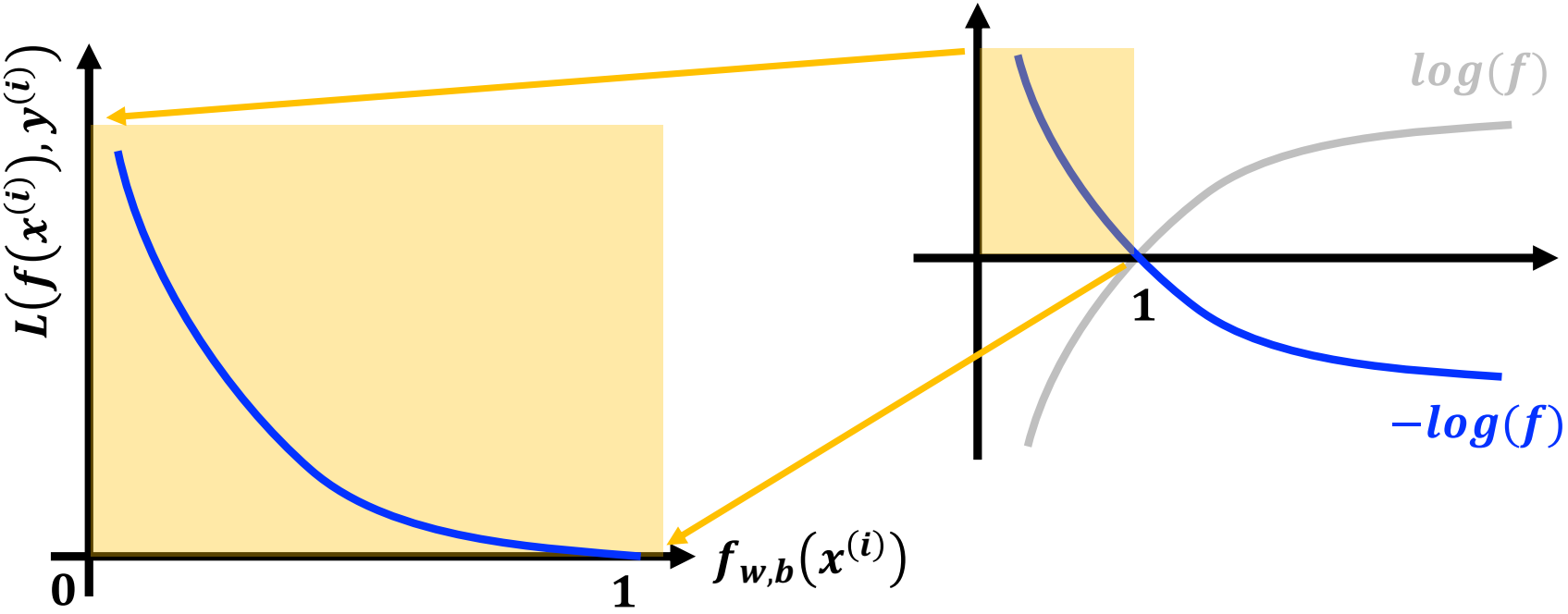
minimize $J(w)$



Choose w to minimize $J(w)$

Loss function

$$L\left(f\left(x^{(i)}\right), y^{(i)}\right) = \begin{cases} -log\left(f_{w,b}\left(x^{(i)}\right)\right) & if\ y^{(i)} = 1 \\ -log\left(1 - f_{w,b}\left(x^{(i)}\right)\right) & if\ y^{(i)} = 0 \end{cases}$$

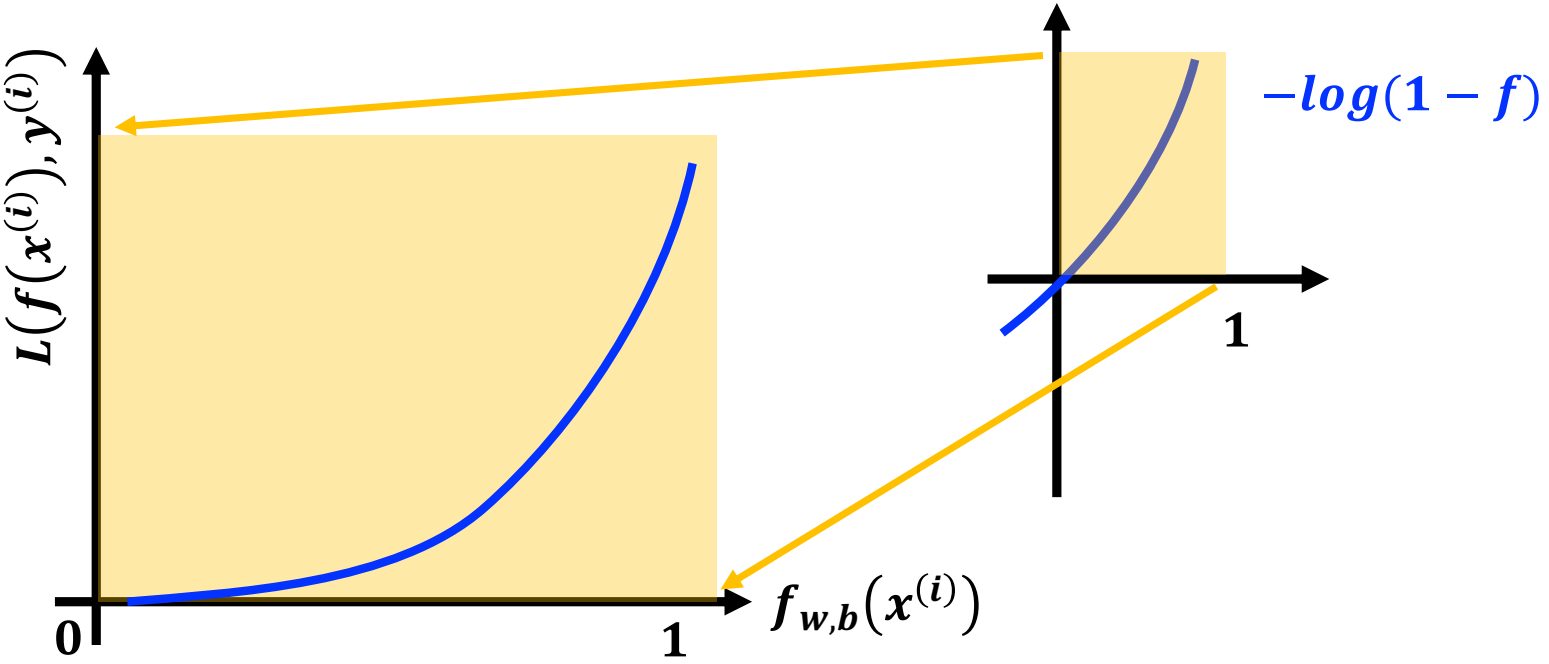


As $f_{w,b}\left(x^{(i)}\right)$ → 1 then loss → 0

As $f_{w,b}\left(x^{(i)}\right)$ → 0 then loss → infinite

Loss function

$$L\left(f\left(x^{(i)}\right), y^{(i)}\right) = \begin{cases} -log\left(f_{w,b}\left(x^{(i)}\right)\right) & if\ y^{(i)} = 1 \\ -log\left(1 - f_{w,b}\left(x^{(i)}\right)\right) & if\ y^{(i)} = 0 \end{cases}$$



As $f_{w,b}\left(x^{(i)}\right)$ → 1 then loss → infinite

As $f_{w,b}\left(x^{(i)}\right)$ → 0 then loss → 0

Gradient descent
(way to minimize the cost function)

Outline

Have some cost function $J(w, b)$

For linear regression
Or any function

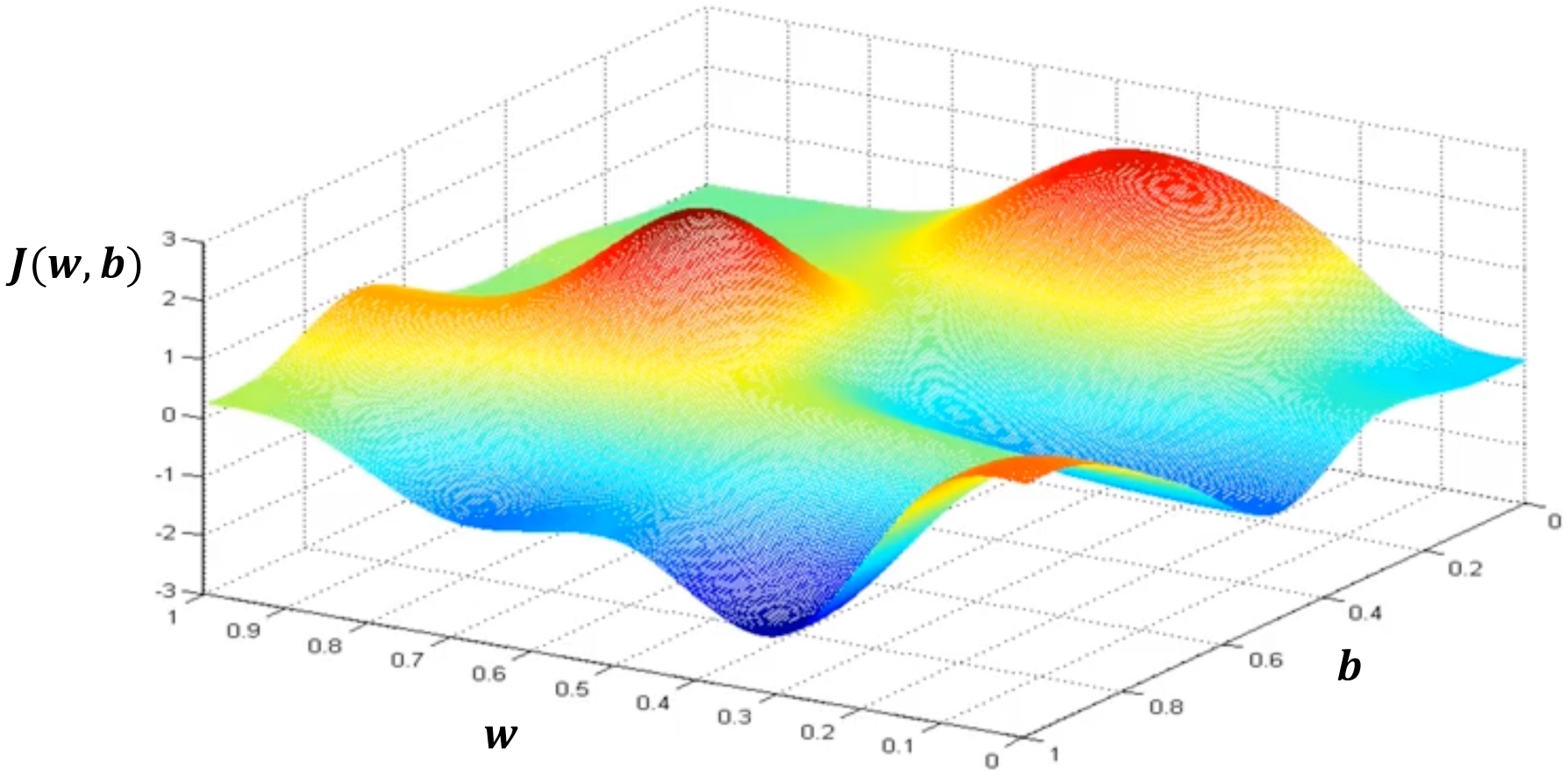
Want $\min_{w,b} J(w, b)$

Outline:

1. start with some w, b (set $w = 0, b = 0$)
2. keep changing w, b to reduce $J(w, b)$
3. until we settle at or near a minimum

Gradient descent

1. Finding way from hill to valley
2. local minimum(s)



Gradient descent algorithm

Repeat until convergence

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b) \qquad b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$

$0 = -\alpha \frac{\partial}{\partial w} J(w,b)$???? → **wrong**

In coding	In math
a = c	a = c
a = a + 1	a = a + 1
(O)	(X)

→ $a_{n+1} = a_n + 1$

Gradient descent algorithm

Repeat until convergence

$$w = w - \alpha \frac{\partial}{\partial w} J(w, b) \qquad b = b - \alpha \frac{\partial}{\partial b} J(w, b)$$

future

present

Learning rate

Partial derivative

Gradient descent algorithm

Repeat until convergence

$$w = w - \alpha \frac{\partial}{\partial w} J(w, b) \qquad b = b - \alpha \frac{\partial}{\partial b} J(w, b)$$

Simultaneously update w and b

Correct

$$tmp\_w = w - \alpha \frac{\partial}{\partial w} J(w, b)$$

$$tmp\_b = b - \alpha \frac{\partial}{\partial b} J(w, b)$$

$$w = tmp\_w$$

$$b = tmp\_b$$

Incorrect

$$tmp\_w = w - \alpha \frac{\partial}{\partial w} J(w, b)$$

$$w = tmp\_w$$

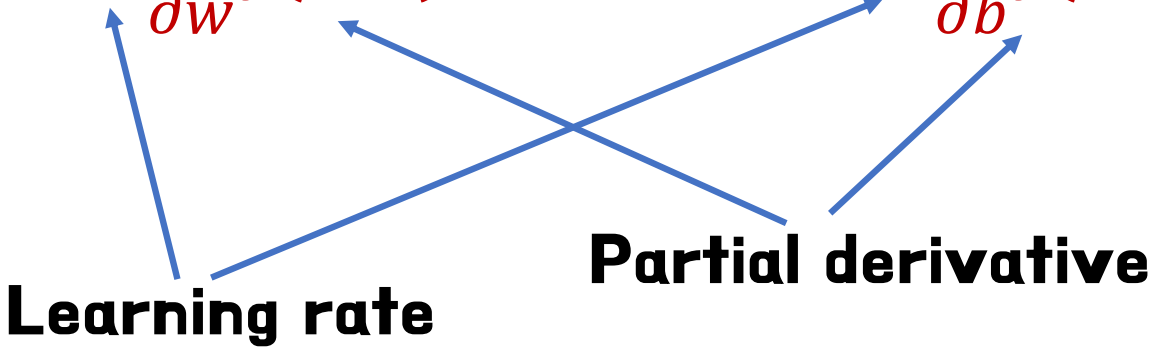
$$tmp\_b = b - \alpha \frac{\partial}{\partial b} J(tmp\_w, b)$$

$$b = tmp\_b$$

Gradient descent algorithm

Repeat until convergence

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b) \qquad b = b - \alpha \frac{\partial}{\partial b} J(w,b)$$



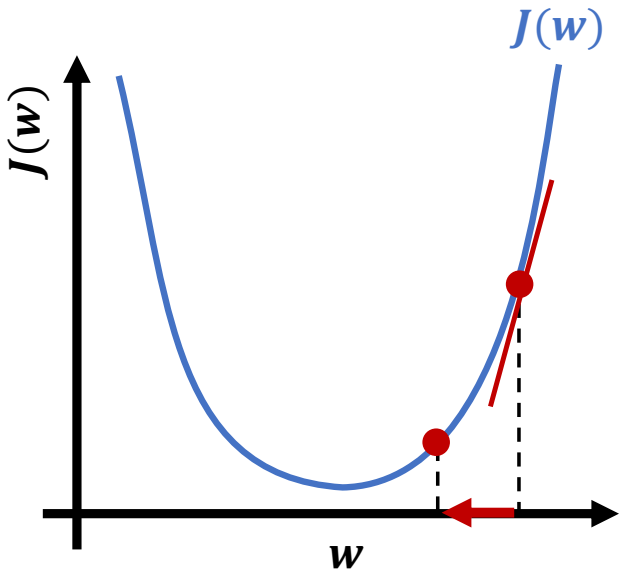
In simplified equation,

$$J(w)$$

$$w = w - \alpha \frac{d}{dw} J(w)$$

$$\min_{w} J(w)$$

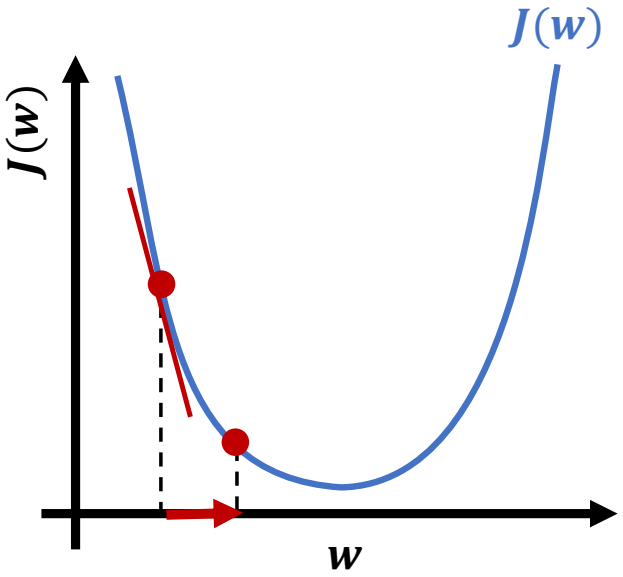
Gradient descent algorithm



$$w = w - \alpha \frac{d}{dw} J(w)$$

$$> 0$$

$$= w - \alpha(positive\ number)$$



$$w = w - \alpha \frac{d}{dw} J(w)$$

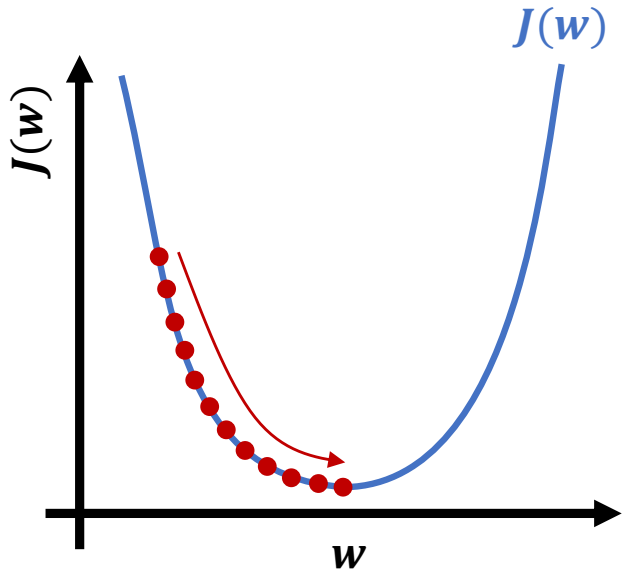
$$< 0$$

$$= w - \alpha(negative\ number)$$

TheMAC

Learning rate

$$w = w - \alpha \frac{d}{dw} J(w)$$



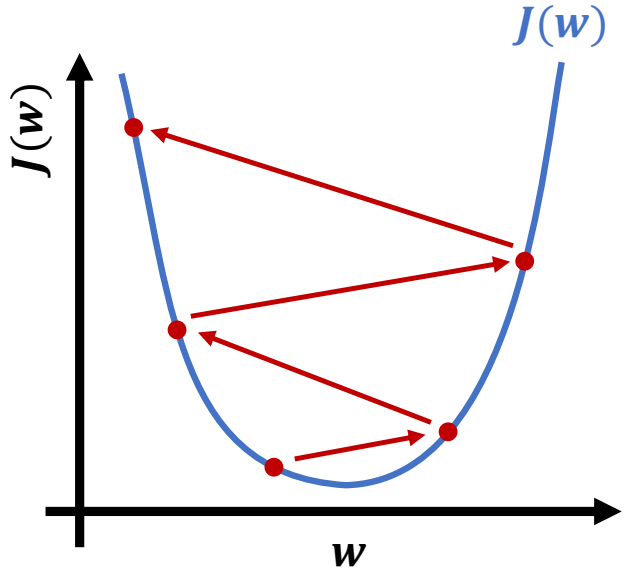
If is α too small …

Gradient descent may be slow

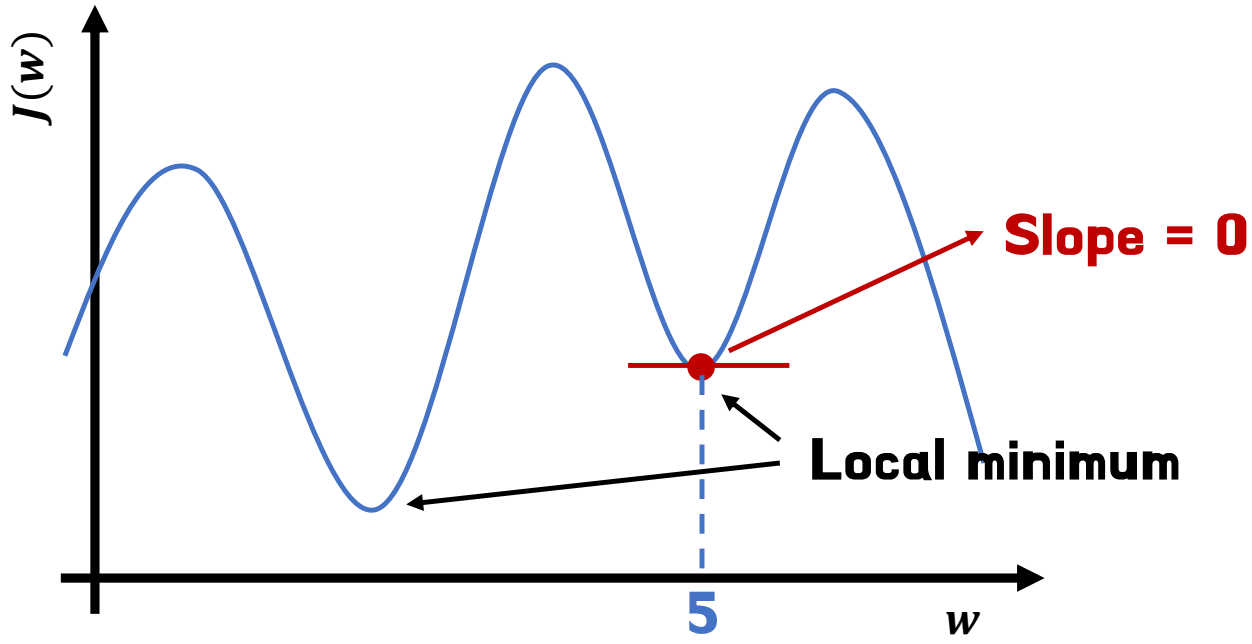
If is α too large …

Gradient descent may:

- **Overshoot, never reach minimum**
- **Fail to converge → diverge**



Learning rate



What happen if the slope is zero ?

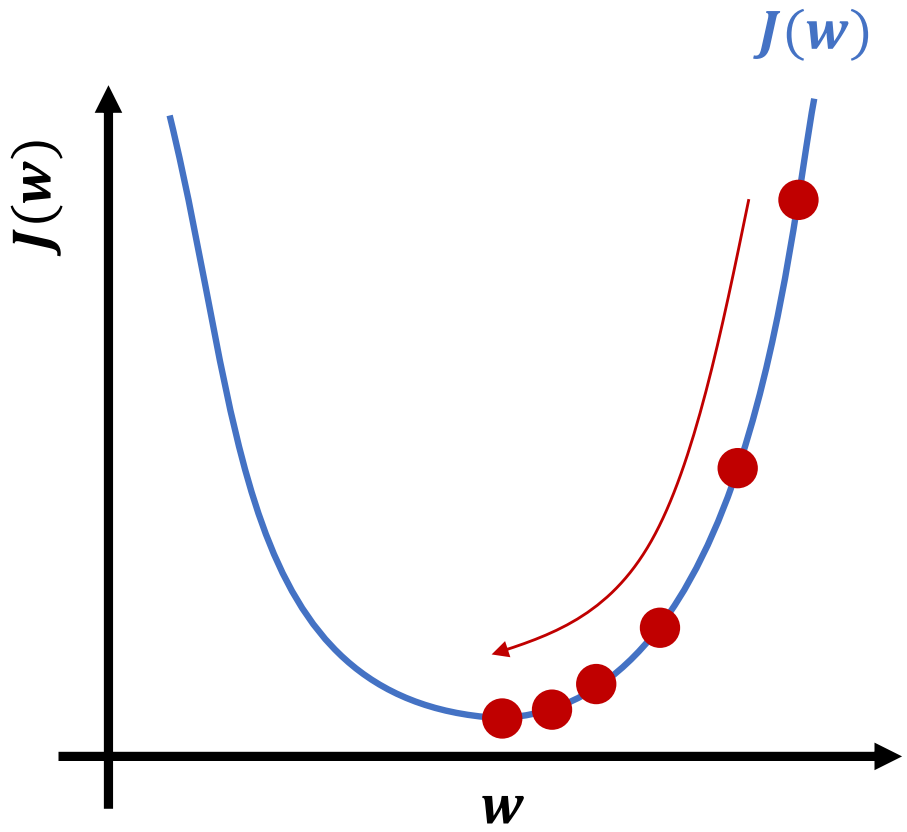
(Let's say α is 0.1)

$$w = w - \alpha \frac{d}{dw} J(w) = 5 - 0.1 * 0 = 5$$ **(no update)**

Learning rate

Can reach local minimum with fixed learning rate ?

$$w = w - \alpha \frac{d}{dw} J(w)$$



Near a local minimum,

- **Derivative → smaller**
- **Update steps → smaller**

Can reach minimum without decreasing learning rate

Gradient descent
(For linear regression)

Gradient descent for linear regression

- **Linear regression model**

$$f_{w,b}(x) = wx + b$$

- **Cost function**

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f(x^{(i)}) - y^{(i)})^2$$

- **Gradient descent algorithm (repeat until convergence)**

$$w = w - \alpha\frac{\partial}{\partial w}J(w,b) \longrightarrow \frac{1}{m}\sum_{i=1}^{m}(f(x^{(i)}) - y^{(i)})x^{(i)}$$

$$b = b - \alpha\frac{\partial}{\partial b}J(w,b) \longrightarrow \frac{1}{m}\sum_{i=1}^{m}(f(x^{(i)}) - y^{(i)})$$

Gradient descent for linear regression

$$\frac{\partial}{\partial w}J(w,b) = \frac{\partial}{\partial w}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(f\left(x^{(i)}\right) - y^{(i)}\right)^2\right] = \frac{\partial}{\partial w}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} + b - y^{(i)}\right)^2\right]$$

$$= \left[\frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} + b - y^{(i)}\right)2x^{(i)}\right] = \frac{1}{m}\sum_{i=1}^{m}\left(f(x^{(i)}) - y^{(i)}\right)x^{(i)}$$

$$\frac{\partial}{\partial b}J(w,b) = \frac{\partial}{\partial b}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(f\left(x^{(i)}\right) - y^{(i)}\right)^2\right] = \frac{\partial}{\partial b}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} + b - y^{(i)}\right)^2\right]$$

$$= \left[\frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} + b - y^{(i)}\right)2\right] = \frac{1}{m}\sum_{i=1}^{m}\left(f(x^{(i)}) - y^{(i)}\right)$$

Gradient descent for linear regression

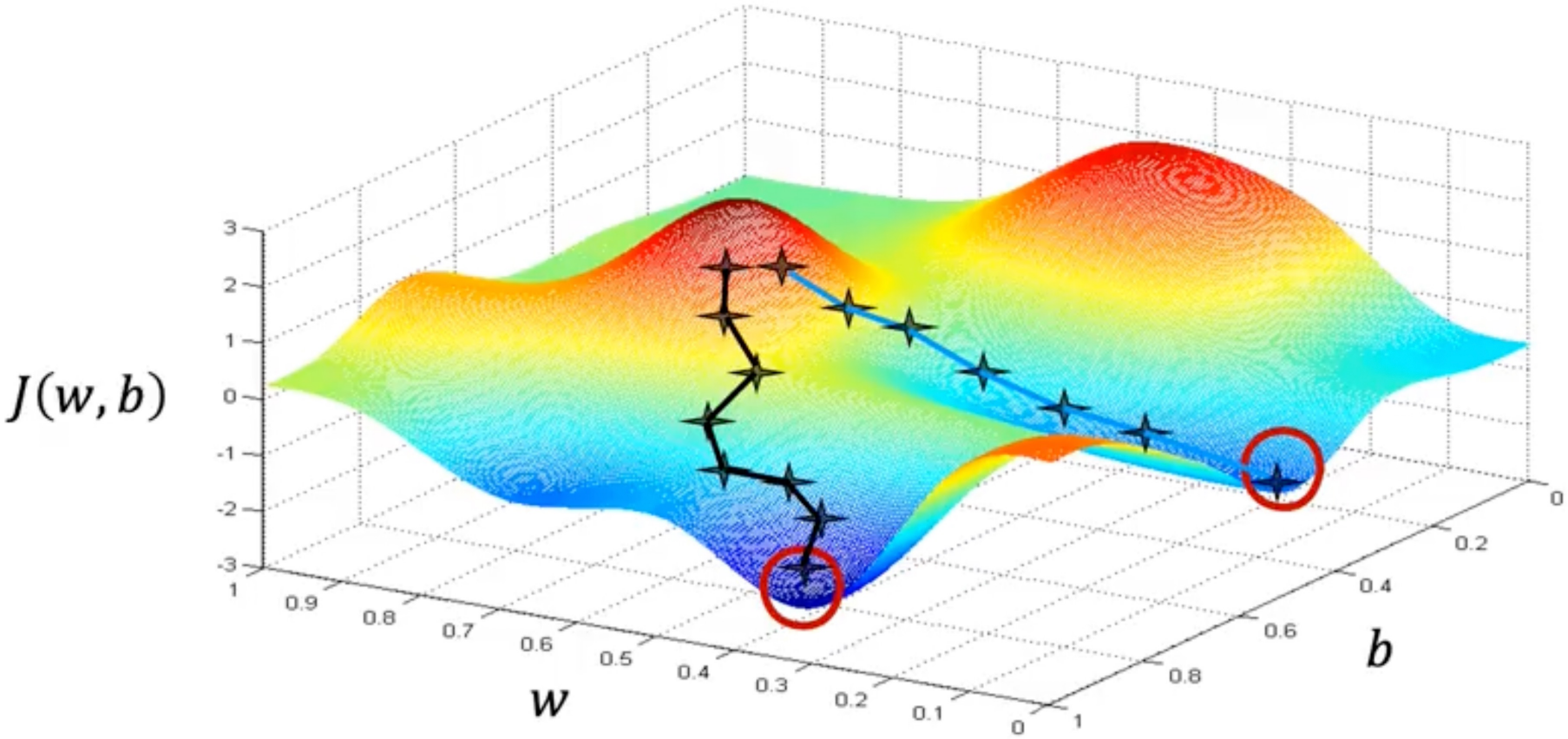
- **repeat until convergence by updating w and b in linear regression**

$$w = w - \alpha \frac{1}{m} \sum_{i=1}^{m} (wx^{(i)} + b - y^{(i)}) x^{(i)}$$

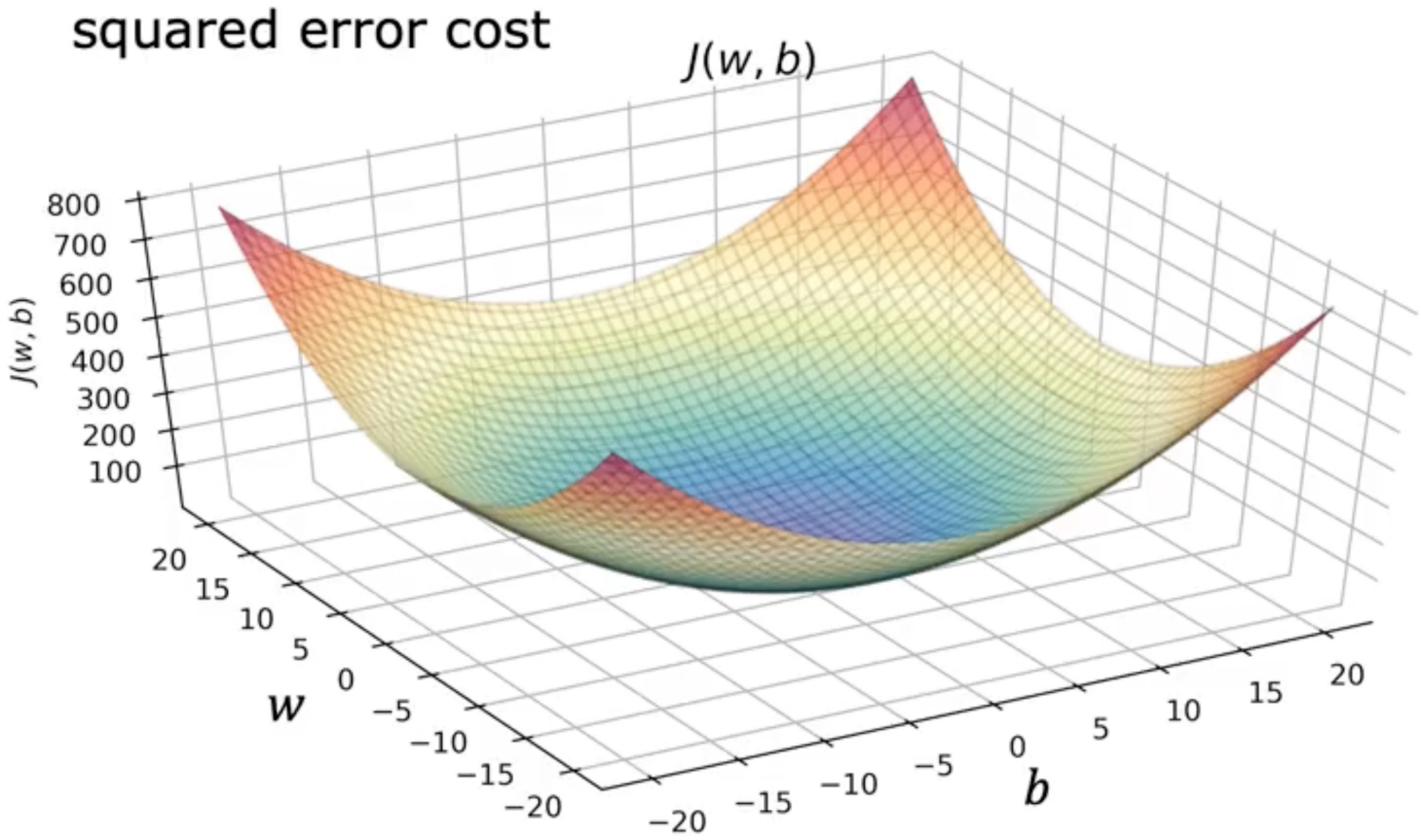
$$b = b - \alpha \frac{1}{m} \sum_{i=1}^{m} (wx^{(i)} + b - y^{(i)})$$

Non-linear cost function

Multiple local minimums



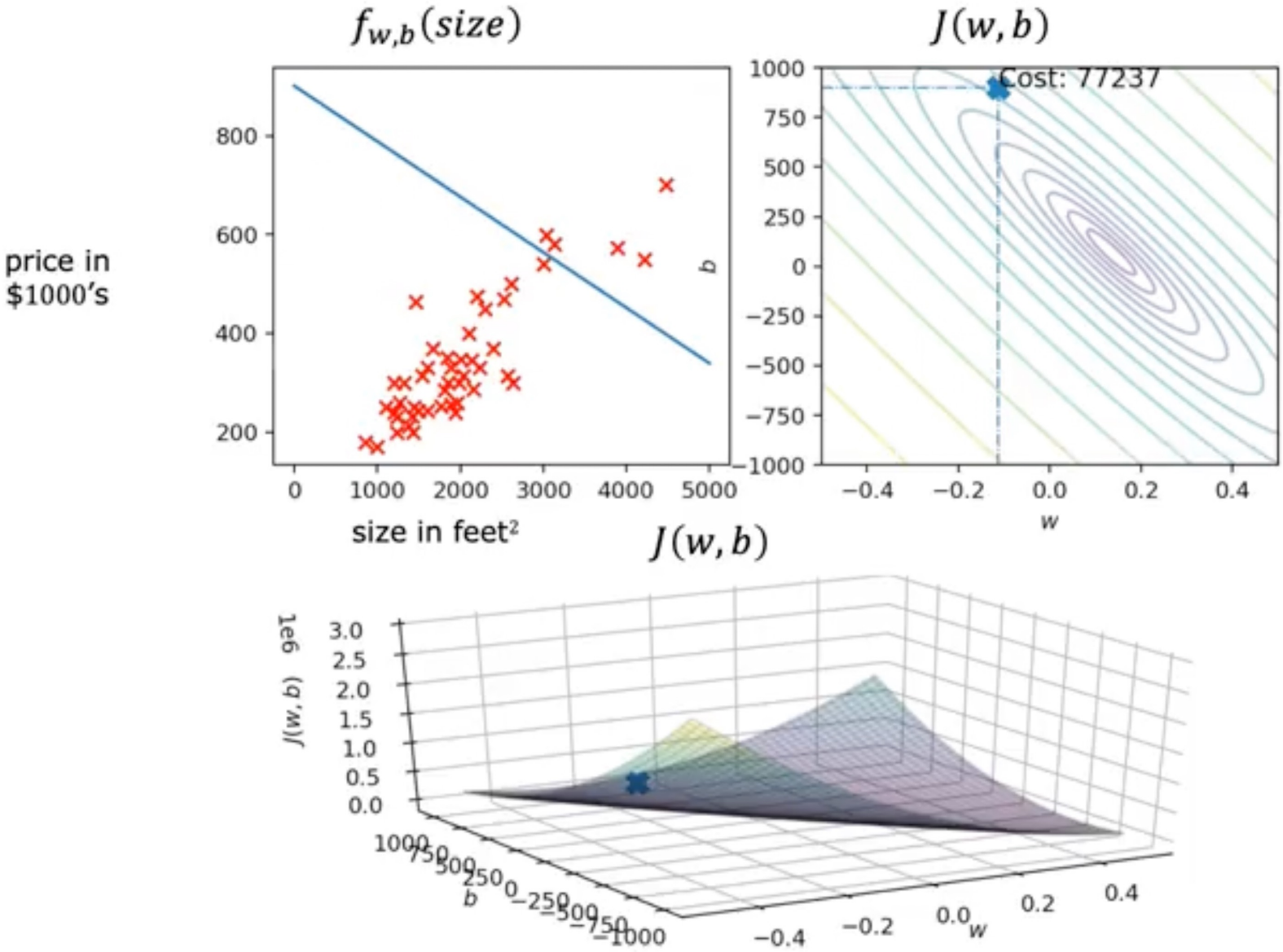
Squared error cost function



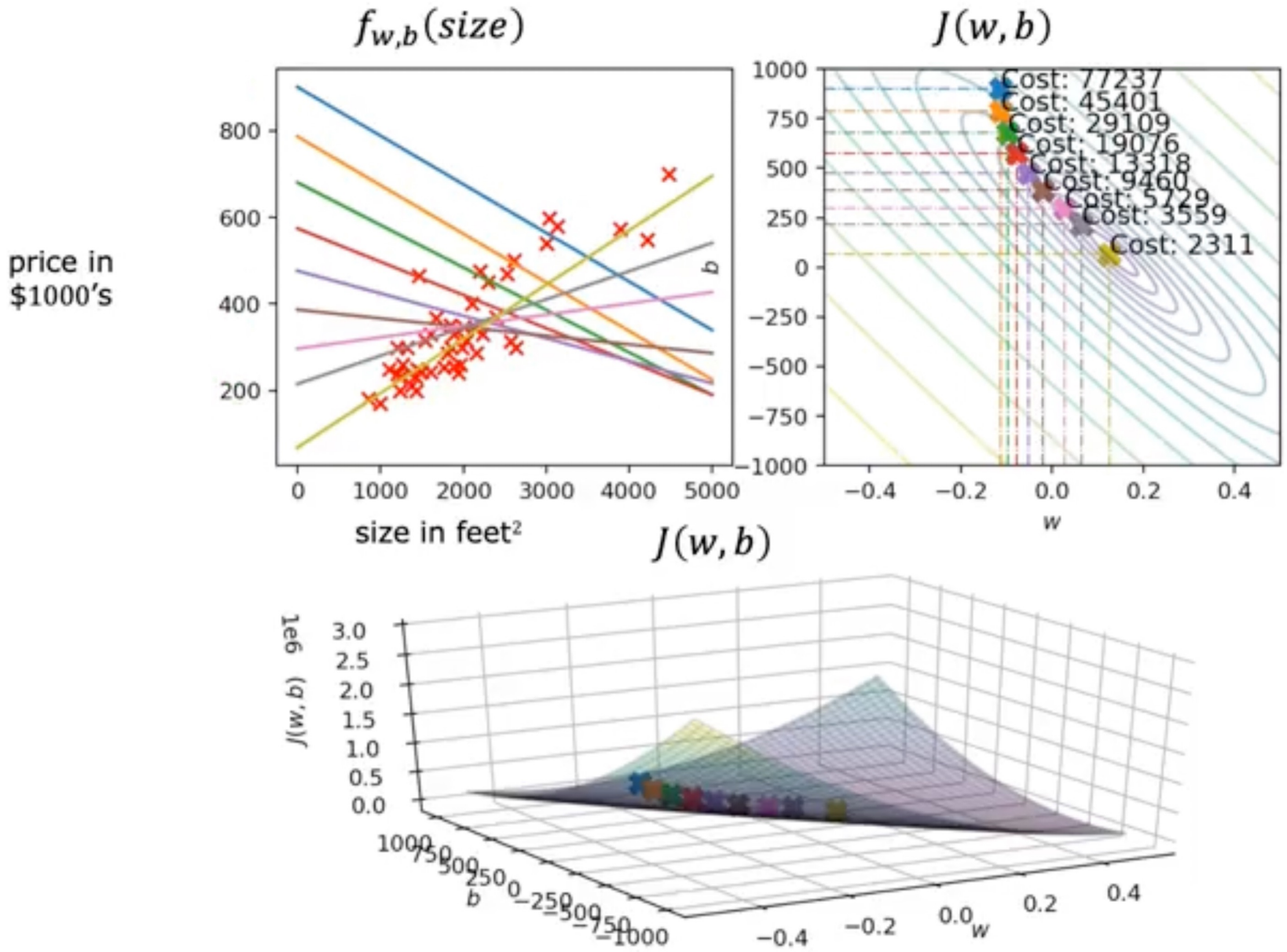
One global minimum in convex function

If you choose proper learning rate, it will always converge to global minimum

Running gradient descent



Running gradient descent



"Batch" gradient descent

"Batch": each step of gradient descent uses all the training examples

	x Size in feet^2	y Price in \$1000's
(1)	2104	400
(2)	1416	232
(3)	1534	315
(4)	852	178
…	…	…
(47)	3210	870

$$m = 47$$

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(f(x^{(i)}) - y^{(i)})^2$$

Gradient descent
(For logistic regression)

Training logistic regression

Find w, b

Given new x, output $f_{w,b}(x) = \frac{1}{1+e^{-(wx+b)}}$

$$P(y = 1|x; w, b)$$

Gradient descent for logistic regression

$$J(w,b) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} log\left(f_{w,b}\left(x^{(i)}\right)\right) + (1- y^{(i)})\, log(1 - f_{w,b}\left(x^{(i)}\right))$$

Repeat until convergence (simultaneous w, b update)

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b) \longrightarrow \frac{\partial}{\partial w} J(w,b) = \frac{1}{m}\sum_{i=1}^{m} (f(x^{(i)}) - y^{(i)})x^{(i)}$$

$$b = b - \alpha \frac{\partial}{\partial b} J(w,b) \longrightarrow \frac{\partial}{\partial b} J(w,b) = \frac{1}{m}\sum_{i=1}^{m} (f(x^{(i)}) - y^{(i)})$$

Not proving this one

Gradient descent for logistic regression

$$w = w - \alpha \frac{1}{m} \sum_{i=1}^{m} (f(x^{(i)}) - y^{(i)}) x^{(i)}$$

$$b = b - \alpha \frac{1}{m} \sum_{i=1}^{m} (f(x^{(i)}) - y^{(i)})$$

Looks like linear regression !

Linear regression $f_{w,b}(x) = wx + b$

Logistic regression $f_{w,b}(x) = \frac{1}{1 + e^{-(wx+b)}}$